Multi Home Transfer Learning for Resident Activity Discovery and Recognition

Parisa Rashidi
Washington State University
Pullman, Washington
prashidi@eecs.wsu.edu

Diane J. Cook
Washington State University
Pullman, Washington
cook@eecs.wsu.edu

ABSTRACT

Activity discovery and recognition can provide unprecedented opportunities for health monitoring, automation, energy efficiency and security. Despite all the potential benefits, in practice we are faced with the main challenge of collecting huge amounts of data for each new physical space in order to carry out the conventional activity discovery algorithms. This results in a prolonged deployment in the real world. More importantly, if we ignore what has been learned in previous spaces, we face redundant computational effort and time investment and we miss the insights gained from past experience that can improve the recognition accuracy. To overcome this problem, we propose a method of transferring the knowledge of learned activities from multiple source physical spaces, e.g. home A and B, to a target physical space, e.g. home C. Our method called Multi Home Transfer Learning (MHTL) is based on a location mining method for target activity discovery, a semi-Em framework for activity mapping, and an ensemble method for label assignment. In this paper we introduce the MHTL methodology. To validate our algorithms, we use the data collected in several smart apartments with different physical layouts.

Categories and Subject Descriptors

H.2.8 [**Information Systems**]: DATABASE MANAGEMENT—*Data mining*; I.2.6 [**Computing Methodologies**]: ARTIFICIAL INTELLIGENCE—*Learning*

General Terms

Activity Discovery, Transfer Learning, Smart Environments

1. INTRODUCTION

With remarkable recent progress in computing power, networking, and sensor technology, we are steadily moving into the world of ubiquitous computing where technology recedes into the background of our lives. Using sensor technology combined with the power of data mining and machine learning, many researchers are now working on smart environments which can discover and recognize residents' activities and respond to the needs of the residents in a context aware way [5]. Some of these efforts have been demonstrated in actual physical testbeds such as the CASAS project [21], the MavHome project [6], the Gator Tech Smart House [12], the iDorm [9], and the Georgia Tech Aware Home [1]. A smart environment typically contains many sensors such as motion sensors that provide us with unprecedented opportunities for health monitoring, automation, energy efficiency and security via activity discovery and recognition [25]. For example researchers are recognizing that smart environments can assist with valuable functions in the area of remote health monitoring and intervention by monitoring the daily activities of elderly adults with memory deficiencies and helping them via timely prompts [23].

In response to this recognized need, researchers have designed a variety of approaches for discovering, modeling and recognizing activities. Those methods exploit naive Bayes [2], decision trees [16], Markov models [15], dynamic Bayes networks [13], conditional random fields [18], frequent sequence mining [10] and mixed frequent-periodic sequence mining [20]. The problem with all those approaches is that they do not exploit the knowledge learned in previous spaces in order to discover and recognize activities in a new space. Therefore, for each new space a huge amount of data needs to be collected in order to carry out the conventional unsupervised activity discovery methods such as frequent or periodic data mining methods. Even if supervised methods are used, a greater burden is placed on the user of the smart environment, who must annotate sufficient data in order to train the recognition algorithms. Our testbeds have required at least one hour of an expert's time to annotate a single day's worth of sensor data. This particularly becomes problematic if we are targeting a deployment in the home of an older adult. Also, learning the model of each environment separately and ignoring what has been learned in other physical settings leads to redundant computational effort, excessive time investment, and loss of beneficial information that can improve the recognition accuracy. Therefore it is beneficial to develop models that can exploit the knowledge of learned activities by employing them in new spaces. This transfer concept results in reducing the need for data collection, reducing or eliminating the need for data annotation and besides achieving an accelerated learning pace. Using multiple sources and fusing their data together can leverage this learning process even more by using a more diverse set of activity models that can help in discovering and recognizing

the target activities.

The process of exploiting the knowledge gained in one problem and applying it to a different but related problem is called transfer learning [19][4]. It is a hallmark of human intelligence, and has been vastly studied in the literature [17], but it has been applied to activity discovery and recognition in very few cases.

Our goal is to transfer the knowledge of learned activities from multiple physical source spaces, e.g. home A and B, to a target physical space, e.g. home C. Previously we have shown a method for transferring learned activities from one resident to another [22]. Zhang et al. [26] have developed a model for mapping different types of activities to each other (e.g. sweeping to cleaning) by learning a similarity function via a Web search. Kasteren et. al [24] describe a simple method for transferring the transition probabilities of Markov models for two different spaces. They only transfer the transition probabilities, and most other activity features such as the activity's structure and related temporal features is ignored, as they assume the structure of HMMs is given and pre-defined.

In our approach, the activity model includes much more information based on using structural, temporal and spatial features of the activities. Also, unlike the approach of Kasteren et al.[24], we do not manually map the sensor networks. Instead, we learn sensor mappings based on the available data and activity models. It should be noted that in order to exploit the knowledge learned in different spaces, we transfer the activities from multiple physical source spaces to a target physical space. First we use a location based data mining method to find target activities in the target data. Then the activities from both source and target spaces are represented in a canonical form called an "activity template" in order to allow for a more efficient mapping process. Next we use a semi-EM framework to map source activities from each single source to the target activities. Finally by using an ensemble learning method based on a weighted majority voting [8] and fusing multiple data sources we assign activity labels to the target activities.

The remainder of the paper is organized as follows. First we describe our approach in more detail, including its three main stages. The first stage discovers activities by mining data and extracting activity models, while the second stage maps source activities to the target activities, and the third stage assigns labels to the target activities. We then show the results of our experiments on data obtained from five different smart apartments. Finally we end the paper with our conclusions and discussion of future work.

2. MODEL DESCRIPTION

Our objective is to develop a method that can transfer learned activities across different physical spaces. We assume that labeled activity data is available in the source space $\mathbb{S}$ consisting of N individual sources $S_1, ..S_N$, and limited unlabeled data is available in the target space T. Our goal is to use the source space knowledge to learn the activity labels in the target space where the physical aspects of the space and the sensors may be different. We assume that the nature of the problem is "inductive transfer learning" or "self taught" [17], i.e. we have labeled data in the source domain, and none or few data labels are available in the target domain. This allows us to reduce several weeks or months of data collection and annotation in the target space to only a few days' worth of data collection. We also assume that the number of available sources (N) is limited and computationally manageable, as reducing the number of sources and source selection is outside the scope of this paper. Our ultimate objective is to be able to correctly recognize the activities in the target space. By using our method, labeled target activity data becomes available that can be consumed by conventional learning algorithms to perform activity recognition, or it can be used as a baseline for other techniques such as active learning techniques. In the remainder of this section we describe our notations and also we will provide a high level description of the algorithm.

Timestamp (ts)	**Sensor ID** (s)	**Label** (l)
7/17/2009 09:52:25	$M004$	Personal Hygiene
7/17/2009 09:56:55	$M030$	Personal Hygiene
7/17/2009 14:12:20	$M015$	None

Table 1: Example sensor data. Here $M004$, $M030$ and $M015$ denote sensor IDs.

The input data is a sequence of sensor events e in the form $e = \langle ts, s, l \rangle$ where ts denotes a timestamp, s denotes a sensor ID, and l is the activity label, if available. An example showing several sensor events can be seen in Table 1. As depicted in Table 1, each sensor event can be part of a labeled activity such as the first and second sensor events, or it can have no activity labels such as the third sensor event. Each sensor ID is associated with its room name (e.g. kitchen) which we will refer to as a location tag L. A standard set of location tags is used across all different sources. We define an activity as $a = \langle \mathcal{E}, l, t, d, \mathcal{L} \rangle$ where $\mathcal{E}$ is a sequence of n sensor events $\langle e_1, e_2, ..e_n \rangle$, l is its label (if available), t and d are the start time and duration of the activity, and $\mathcal{L}$ represents the set of location tags where a has occurred.

We denote the set of activities in each individual source space S_k as $\mathcal{A}_{S_k}$. The set of all source activities is denoted by $\mathcal{A}_{\mathbb{S}}$ which is the union of activities from all individual sources, i.e. $\mathcal{A}_{\mathbb{S}} = \bigcup_k \mathcal{A}_{S_k}$. The set of target activities is denoted by $\mathcal{A}_T$. The set of source sensors and the set of target sensors is denoted by $\mathcal{S}_{\mathbb{S}}$ and $\mathcal{S}_T$. In order to be able to map activities from the source space to a target space, we need to find a way to map the source sensor network to the target sensor network i.e. we're looking for the mapping $\mathcal{F}'(\mathcal{S}_{\mathbb{S}}) = \mathcal{S}_T$, as the source sensors will have different locations and properties than the target sensors. Based on using activity features and also the sensor mappings $\mathcal{F}'$, we will find the activity mapping function $\mathcal{F}(\mathcal{A}_{\mathbb{S}}) = \mathcal{A}_T$. Note that for the individual mappings from S_k to T the above mapping functions are written as $\mathcal{F}_k(A_{S_k})$, and $\mathcal{F}'_k(S_{S_k})$.

The extent to which an activity $a_i \in \mathcal{A}_{S_k}$ maps to activity $a_j \in \mathcal{A}_T$ is reflected in matrix M_k, where $M_k[i, j] \in [0..1]$ shows the probability that activity a_i and a_j have the same label. Similarly, a second matrix $m_k[p, q] \in [0..1]$ shows the probability that sensor $s_p \in \mathcal{S}_{S_k}$ maps to sensor $s_q \in \mathcal{S}_T$

based on their location and their role in activity models. Note that the mappings need not to be one to one, due to the differences in the number of sensors and number of activities in the source and target spaces.

Our multi home transfer learning algorithm (MHTL) performs activity discovery and transfer in several stages (see Figure 1). The first step involves processing labeled data from the source space and mining available unlabeled data from the target space in order to extract the activity models in each space. In the source space, for each individual source S_k we extract the activities $\mathcal{A}_{S_k}$ by converting each contiguous sequence of sensor events with the same label to an activity. To reduce the number of activities and to find a canonical mapping, similar activities in $\mathcal{A}_{S_k}$ are consolidated together to represent an "activity template". To avoid mapping irrelevant sensors, a filter feature selection method based on mutual information [11] is used to remove the irrelevant sensors for each activity template. In the target space the data is mined to find unlabeled activity patterns based on using location closure. Target activities are then consolidated using an incremental clustering method [3]. If any labeled data is available in the target space, it can be used to refine the target activity models.

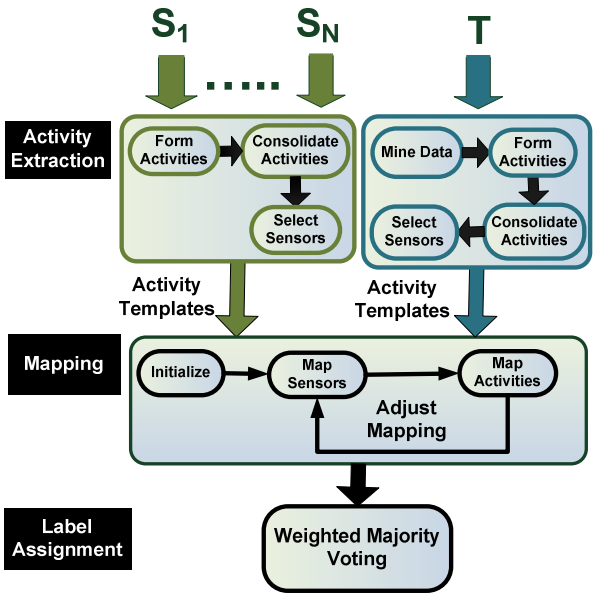


Figure 1: Main components of MHTL for transferring activities from multiple source spaces to a target space.

Next, source activity templates are mapped to the target activity templates. First the activity templates' initial mapping probabilities are computed based on structural, temporal and spatial similarities. The sensors' initial mapping probabilities are assigned based on a spatial similarity measure. After initialization, the algorithm starts an Expectation Maximization like framework [7] called semi-EM in an iterative manner. First, the sensor mapping probabilities are adjusted based on the activity mapping probabilities, next the activity mapping probabilities are adjusted based on the updated sensor mapping probabilities. This continues until no more changes are perceived or until a user defined number of iterations is reached.

Finally, we assign an activity label to each target activity a_j based on the obtained activity mapping probabilities M. To map activity labels we use an ensemble method based on a weighted majority voting. For each space S_k a vote for the label of the target activity a_j is casted. The voted label is selected the same as the label of a source activity a_i that maximizes the mapping probability M_k for a_j. Each vote is weighted by the overall similarity between the source space S_k and the target space T, as will be described later. At the end, the label with the maximum weighted votes is considered as the label of the activity a_j. Note that in this method all the sources contribute to the label mapping process in order to generate a final activity label for each target activity. We provide a more detailed description of these three steps in the following discussion.

2.1 Activity Extraction

The first step of the multi-stage MHTL algorithm is to extract the activity models from input data in both source and target spaces. For each single source space S_k we convert each contiguous sequence of sensor events with the same label to an activity a. This results in finding the set of activities $\mathcal{A}_{S_k}$ for each one of the individual source spaces S_k. The start time of the activity is the timestamp of its first sensor event, while its duration is the difference between its last and first timestamps. Due to the prohibitively large number of extracted activities and possible similarity among them, we combine similar activities together as an "activity template". Representing a set of similar activities as an activity template allows for a more efficient canonical mapping from source to target, as only a few activity templates will be mapped from source to target instead of mapping a large number of similar activities with only minor differences. The activity template for a set of activities is itself an activity, formed by merging activities' sensors, durations, and start times where the merged start times and durations form a mixture normal distribution. The temporal mixture model allows us to capture and model variations of the same activity that occur at different times. For example consider the "eating" activity which usually happens three times a day, once in the morning as breakfast, once at noon as lunch, and once at night as dinner. Using a mixture model for the start time we are able to capture all the three variations by using a single activity model. During activity consolidation, all the source activities that have the same label will be consolidated into one single activity template. Note that as each activity template is itself an activity, we use the terms activity and activity template interchangeably.

The next step after similar activities are consolidated together is to perform sensor selection for each activity template a by preserving only relevant sensors. Performing sensor selection on each activity template allows for even a more compact representation and a more accurate mapping, as it allows us to map only the relevant sensors and to avoid mapping the irrelevant sensors as noise. Our sensor selection method is a filter feature selection method based on mutual information [11]. For each activity template a and each sensor s we define their mutual information $MI(s, a)$ as in Equation 1. It measures their mutual dependence and

shows how relevant is sensor s in predicting the activity's label. Here $P(s,a)$ is the joint probability distribution of s and a, while $P(s)$ and $P(a)$ are the marginal probability distributions, all computed from the sensor and activity occurrences in the data. A high mutual information value indicates the sensor is relevant for the activity template. We simply consider sensors with a mutual information above the midpoint (0.5) as relevant, otherwise they will be discarded.

$$MI(s,a) = P(s,a) * \log \frac{P(s,a)}{P(s)P(a)} \tag{1}$$

To find activity patterns in unlabeled target data, we perform a data mining step on the input data. First we partition the input data into activities. A sensor event $e_1 = \langle ts_1, s_1, l_1 \rangle$ and a successor sensor event $e_2 = \langle ts_2, s_2, l_2 \rangle$ are part of the same activity if $L_{s_1} = L_{s_2}$, i.e. if both sensors are in the same location. Such a local partitioning allows us to have a baseline for finding individual activities. This approach is based on the intuition that occurrences of the same activity usually happen within the same location (such as preparing meal in the kitchen, grooming in the bathroom, etc), and more complex activities occurring in different locations can be composed of those basic activities. Notice that as we only have access to limited input data (perhaps a few days or even a few hours), we cannot use conventional activity discovery methods such as frequent or periodic sequence mining methods [20] to find activity patterns in the data. Therefore exploiting the spatial closure can be a way to overcome this problem. After partitioning data into the initial activities, we consolidate those activities by grouping together similar activities into an activity template. To combine activities together, we use an incremental clustering method [3], such that each activity is assigned to the most similar centroid if their similarity is above threshold ς, and then the centroid is recomputed. Otherwise the activity forms a separate cluster. The centroid is itself represented as an activity template. At the end all the activities in one cluster are consolidated together and the sensor selection is carried out. For two activities a_i and a_j, their similarity $\Upsilon(i,j)$ is defined as in Equation 2.

$$\Upsilon(i,j) = \Upsilon_t[i,j] + \Upsilon_d[i,j] + \Upsilon_{\mathcal{L}}[i,j] + \Upsilon_S[i,j] \tag{2}$$

In above equation, Υ_t refers to start time mapping (if the two activities happen at similar times, e.g. both around noon), Υ_d refers to duration mapping (if the two activities have similar durations), $\Upsilon_{\mathcal{L}}$ refers to location mapping (if the two activities happen in similar locations, e.g. both in the kitchen), and Υ_S refers to structure mapping (if the two activities have similar structure in terms of sensors). We normalize $\Upsilon(i,j)$ to fall within the range [0..1]. For simplicity, we have chosen the mappings to have equal effects, however it's possible to define $\Upsilon(i,j)$ as a weighted average.

As mentioned, the start times are in form of a mixture normal distribution with means $\Theta = \langle \theta_1..\theta_r \rangle$. We represent start time θ in an angular form Φ measured in radians instead of a linear representation. This allows for time differences to be represented correctly (2:00 am will be closer to 12:00 pm rather than 5:00 am). Then the similarity between the two start time distributions will be as in Equation 3.

$$\Upsilon_t[i,j] = \max_{\substack{\theta_1 \in \Theta_i \\ \theta_2 \in \Theta_j}} (1 - \frac{|\Phi_{\theta_2} - \Phi_{\theta_1}|}{2\pi}) \tag{3}$$

Duration mapping is calculated as in Equation 4 where durations are in form of a mixture normal distribution with means $\Gamma = \langle \gamma_1..\gamma_r \rangle$.

$$\Upsilon_d[i,j] = \max_{\substack{\gamma_1 \in \Gamma_i \\ \gamma_2 \in \Gamma_j}} (1 - \frac{|\gamma_2 - \gamma_1|}{\max(\gamma_2, \gamma_1)}) \tag{4}$$

To compute $\Upsilon_{\mathcal{L}}$ we use Equation 5 which is the Jaccard similarity coefficient for the sets of locations of the two activities. A similar Jaccard similarity coefficient based on similar sensors is defined for the structure mapping Υ_S in Equation 6.

$$\Upsilon_{\mathcal{L}}[i,j] = \frac{\mid \mathcal{L}_i \bigcap \mathcal{L}_j \mid}{\mid \mathcal{L}_i \bigcup \mathcal{L}_j \mid} \tag{5}$$

$$\Upsilon_S[i,j] = \frac{\mid \mathcal{E}_i \bigcap \mathcal{E}_j \mid}{\mid \mathcal{E}_i \bigcup \mathcal{E}_j \mid} \tag{6}$$

2.2 Mapping Activities

The next step after the activity models for the source and target space have been identified is to map the source activity templates to the target activity template. First we initialize the sensor and activity mapping matrixes, m_k and M_k for each pair of source S_k and target T. The initial values of the sensor mapping matrix $m_k[p,q]$ for two sensors $s_p \in S_k$ and $s_q \in T$ is defined as 1.0 if they have the same location tag, and as 0 if they have different location tags. The initial value of $M_k[i,j]$ for two activities $a_i \in \mathcal{A}_{S_k}$ and $a_j \in \mathcal{A}_T$ is obtained based on exploiting related spatial and temporal information and also prior activity label information (if available), as in Equation 7. Note that in Equation 7 the first case applies to the few labeled target activities, while for the majority of the target activities the second case is applied.

$$M_k[i,j] = \begin{cases} 1.0 & if \quad l_i = l_j \\ \Upsilon(i,j) & otherwise \end{cases} \tag{7}$$

For computing subsequent mapping probabilities, we use an Expectation Maximization (EM) like framework [7] by estimating the mapping probabilities in an iterative manner. First, the sensor mapping probabilities are computed; and in the next step the activity mapping probabilities are maximized based on the sensor probabilities. Though this model doesn't exactly reflect an EM algorithm, however due to its iterative manner and likelihood estimation in two steps, we refer to it as a semi-EM framework.

To compute sensor mapping probabilities $m_k[p,q]$ for sensors $s_p \in \mathcal{S}_{S_k}$ and $s_q \in \mathcal{S}_T$, we rely on activities in which s_p and s_q appear in, as in Equation 8. The learning rate α refers to how fast we want to converge on the new values, while $m_k^n[p,q]$ and $m_k^{n+1}[p,q]$ refer to the current and updated values of $m_k[p,q]$ in iteration n and $n+1$, respectively.

$$m_k^{n+1}[p,q] = m_k^n[p,q] - \alpha * \Delta m_k[p,q] \tag{8}$$

$$\Delta m_k[p,q] = m_k^n[p,q] - \frac{1}{|X_p||Y_q|} \sum_{a_i \in X_p} \sum_{a_j \in Y_q} M_k[i,j] \tag{9}$$

$$\begin{aligned} X_p &= \{a_i \in \mathcal{A}_{S_k} | s_p \in \mathcal{E}_i\} \\ Y_q &= \{a_j \in \mathcal{A}_T | s_q \in \mathcal{E}_j\} \end{aligned} \tag{10}$$

In Equation 9, X_p and Y_q for sensor p and q give us all the activities in which the sensors appear. This means that those activities which do not include a given sensor will not contribute to that sensor's mapping probability.

In the next step, to adjust the mapping probability between each two activities, we use Equation 11 to account for the updated sensor mappings. Here $M_k^n[i,j]$ and $M_k^{n+1}[i,j]$ refer to the current and updated values of $M_k[i,j]$ in iteration n and $n+1$, respectively.

$$M_k^{n+1}[i,j] = M_k^n[i,j] - \alpha * \Delta M_k[i,j] \tag{11}$$

$$\Delta M_k[i,j] = M_k^n[i,j] - \frac{1}{|\mathcal{E}_i|} \sum_{s_p \in \mathcal{E}_i} \max_{\substack{q \\ s_q \in \mathcal{E}_j}} m_k[p,q] \tag{12}$$

The above procedure for computing sensor mapping and activity mapping probabilities is repeated until no more changes are perceived or until a pre-defined number of iterations is reached. Next, the labels are assigned to the target activities based on the obtained probability mapping matrices.

2.3 Mapping Labels

In order to assign labels to the target activities, we use a voting ensemble method [8] based on the activity models $\mathcal{A}_{S_k}$ for each space S_k. Combining data from different sources to improve the accuracy and to have access to complimentary information is known as data fusion or as a form of ensemble learning [14]. Ensemble learning is a strategic way to combine multiple models, such as different classifiers or hypotheses to solve a computational problem. In our problem, using multiple sources allows us to fuse data from different sources and form different activity models, therefore being able to map target activities based on a more diverse set of source activities. In order to be able to successfully apply the ensemble learning technique, an ensemble system needs classifiers whose decision boundaries are adequately different from each other. The most popular method is to use different training datasets to train individual classifiers. The diversity condition of ensemble learning in our problem is achieved by using different training sets from N different physical source spaces, resulting in N different hypotheses. We build a classifier based on each individual hypothesis h_k and then by combining the predicted labels of all classifiers for a certain target activity we are able to make a decision about the activity's final label.

Data: $\mathcal{A}_{\mathbb{S}}$, M, m, a_j
Result: l_{a_j}
`// The voted labels and their weights`
Λ, W
foreach $S_k \in \mathbb{S}$ **do**
  $l = l_{a_i} \quad s.t. \quad M_k[i,j] = \max_z(M_k[z,j])$
  add l to Λ as $\Lambda[l]$
  $\mathrm{Sim}(S_k, T) = \sum_{a_x \in \mathcal{A}_{S_k}} M_k[x, \mathcal{F}_k(a_x)]$
  $W[l] = W[l] + \frac{Sim(S_k,T)}{|\mathcal{A}_{S_k}|}$
end
$l_{a_j} = \max_l W[l]$
return a_j

Figure 2: The weighted majority voting schema for label assignment.

Each hypothesis h_k is constructed based on using the activity templates $\mathcal{A}_{S_k}$ for space S_k plus the activity and sensor mapping probabilities M_k and m_k. We represent each hypothesis as $h_k = \{\mathcal{F}_k, \mathcal{F}'_k\}$ where $\mathcal{F}$ and $\mathcal{F}'$ denote the activity and sensor mapping functions. For a single space S_k, Equations 13, 14 and 15 provide us with the activity mapping function $\mathcal{F}$, sensor mapping function $\mathcal{F}'$ and the assigned label l_{a_j} for each activity $a_j \in T$. As can be seen in Equation 15, the target activity's label is selected to be the same as the label of a source activity $a_i \in S_k$ that maximizes the mapping probability M_k for a_j.

$$\mathcal{F}_k(a_i) = \max_{a_j}(M_k[i,j]) \tag{13}$$

$$\mathcal{F}'_k(s_p) = \max_{s_q}(m_k[p,q]) \tag{14}$$

$$l_{a_j} = l_{a_i} \quad s.t. \quad M_k[i,j] = \max_z(M_k[z,j]) \tag{15}$$

In order to combine the assigned labels for each a_j using different hypotheses, we use the weighted majority voting algorithm as in Figure 2.3. The input of this algorithm is the source activities $\mathcal{A}_{\mathbb{S}}$, the activity mapping probabilities M, the sensor mapping probabilities m, and activity a_j. The output of the algorithm is the label of a_j as l_{a_j}. For each source space S_k we find the label of a_j by using Equation 15. Each predicted label l is associated with a weight $W[l]$,

which is the total similarity between the source S_k and T. The total similarity between S_k and T is calculated as in Equation 16 by summing over the best mapping from S_k to T for each $a_i \in S_k$. Obviously a label can be voted for by different hypotheses and its weight will be increased as a result.

$$Sim(S_k, T) = \sum_{a_x \in \mathcal{A}_{S_k}} M_k[x, \mathcal{F}_k(a_x)] \tag{16}$$

At the end, the label with the greatest number of weighted votes is considered as the label of the activity a_j. After obtaining the labels of all target activities, we can use the obtained labels to train a conventional activity recognition algorithm. We also can use the labels in conjunction with other techniques such as active learning in order to further improve the results.

3. EXPERIMENTS

We evaluated the performance of our MHTL algorithm using the data collected from five different smart apartments. The layout of the apartments including sensor placement and location tags are shown in Figure 3. We will refer to apartments in Figures 3(a) through 3(e) as apartments 1 to 5. The data was collected during a three month period for apartments 1, 2, and 3, and during a two month period for apartments 4 and 5. Each apartment is equipped with motion sensors, and most of the apartments are also equipped with contact sensors which monitor the open/closed status of doors and cabinets. Apartment 5 is also equipped with light sensors and some item sensors to sense the presence of key items. As can be seen in Figure 3 the apartments have different layouts. For example, apartments 3 and 4 have two bedrooms, while apartments 1 and 2 have one bedroom. In addition, some functional spaces might not be available in all five apartments, such as the workspace, laundry room or the music room.

The residents also have quite different schedules, as can be seen from the activity distribution diagrams shown in Figure 4. For example, in the first apartment housekeeping is performed each Friday, while in the second apartment this is performed once a month, and in the third apartment the housekeeping activity is replaced by a work activity. Also the activity level in each apartment is different, as can be seen clearly by comparing activity distribution diagrams for apartment 4 versus other apartments. The activity level is dependent on the activity level of the residents as well as the number of sensors that monitor the activities. The three first apartments were single resident apartments, while for the fourth apartment the residents included a man, a woman, and a dog. The fifth apartment included two undergraduate student residents. All the data was collected while residents were performing their normal daily activities during a three/two month period.

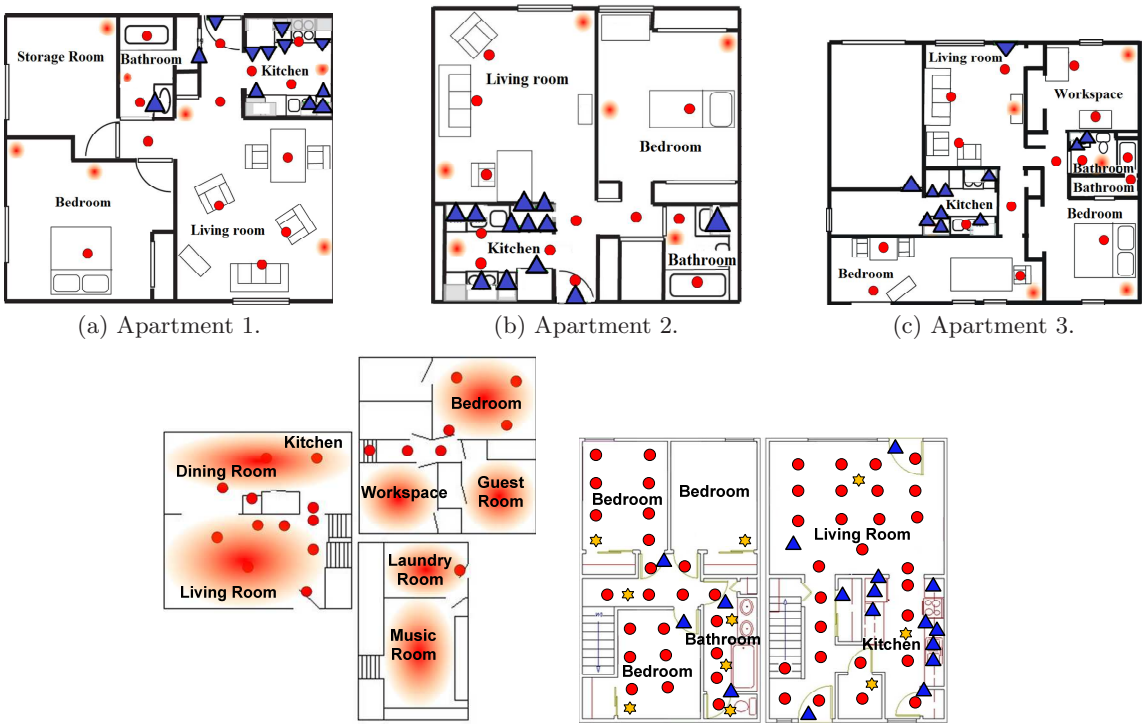
Each of the datasets was annotated with activities of interest for the corresponding resident and apartment. A total of 11 activities were noted for apartments 1, 2 and 3. Those activities included bathing, bed-toilet transition, eating, enter home, housekeeping (for the third apartment this is replaced by "work"), leave home, meal preparation, personal hygiene, sleeping in bed, sleeping not in bed (relaxing) and taking medicine. For the fourth apartment, 7 activities were noted including bed-toilet transition, taking medicine, eating, leaving home, laundry, sleeping in bed and working. The fifth apartment included 7 activities as working, sleeping in bed, bed-toilet transition, personal hygiene, meal preparation, housekeeping, sleeping not in bed (relaxing).

We ran our algorithm for each one of the apartments as the target space, resulting in five different transfer learning problems. In each setting, all the apartments except for the target apartment were used as the source apartments. In each setting, we used all the available source labeled data, 1 to 7 days of target unlabeled data, and 0 to 1 days of target labeled data.

The first step, activity extraction, resulted in a considerable reduction in the number of source activities. In particular 3384, 2602, 1032, 428, and 492 activity instances from the first, second, third, fourth and fifth apartments were represented by as few as 11, 10, 9, 7, and 7 activity templates. The reason that we have obtained less templates than the 11 predefined activities in the second and third apartment is that the "eating" activity was done rather in an erratic way and in different locations, therefore our sensor selection algorithm didn't choose any specific sensor for that activity, and as a result the activity was eliminated. The same applied for "taking medicines" in third apartment. This shows how our algorithm can avoid mapping very irregular activities. It also shows how the algorithm condensed the activity instances into a compressed representation, as we approximately obtained the 11 predefined activities for the first three apartments and exactly 7 activities for the last two apartments. During activity extraction, also the number of sensors for each activity template was reduced from an average of 69.32 sensors to 3.73 sensors, as the algorithm removed the irrelevant sensors and preserved only the relevant sensors. This shows that for each activity a few key sensors can be used to identify the activity, e.g. taking medicine can be identified by the cabinet sensor where the medicines are kept.

In the target space, data was mined to extract the activity templates. For example, using three days of unlabeled target data and no labeled target data, we discovered 8, 7, 7, 5 and 5 activity templates for the apartments 1 through 5. The similarity threshold ς in those experiments was set to the midpoint 0.5. The reason that fewer activity templates are discovered compared to the predefined activities, is because some similar activities might be merged into one activity, such as relaxing and eating which happen at similar times and similar places. Also for some other activities it is not very easy to discover them from only a few days of data, such as housekeeping which happens quite rarely compared to other activities; and even if it happens to be in the data, because of its erratic nature and occurring all over the home, it is not very easy to be discovered.

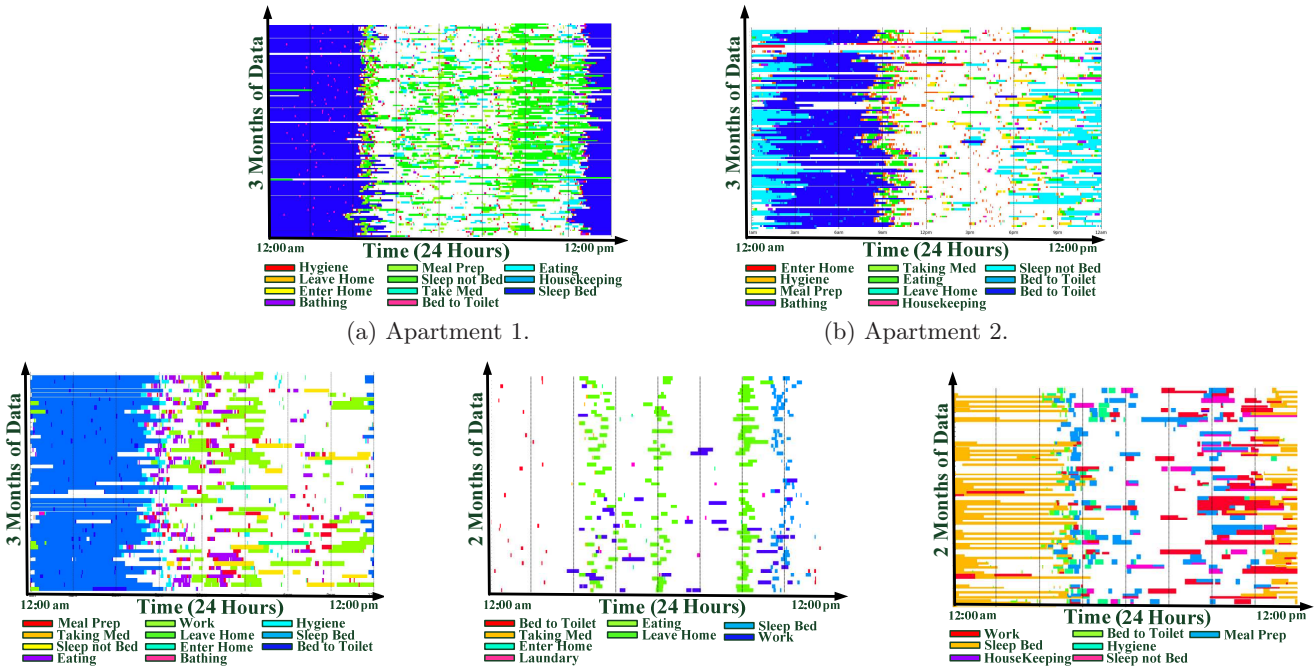
In the next step the source activities were mapped to the target activities. In order to be able to evaluate the mapping accuracy of our algorithm, we embedded the actual labels of target activities in data. This label is not used during train-



(a) Apartment 1. (b) Apartment 2. (c) Apartment 3.

(d) Apartment 4. (e) Apartment 5.

Figure 3: Figures (a-e) show sensor map and location tags for each apartment. On the map, circles show motion sensors while triangles show switch contact sensors. The hollow-shaped motion sensors (as in Figure d) are the area motion sensor. The stars in Figure (e) show the light sensors.



(a) Apartment 1. (b) Apartment 2.

(c) Apartment 3. (d) Apartment 4. (e) Apartment 5.

Figure 4: Figures (a-e) show residents' activity distribution per 24 hour (horizontal axis) for three/two months of data (vertical axis).

ing, rather it's only used at the end to verify the correctness of the results. The mapping accuracy is defined as number of target activities which their assigned label matches the actual embedded label. Figure 6 shows the mapping accuracy for different amounts of unlabeled target data and no labeled target data, in several different settings. Figure 6 also shows a comparison between mapping accuracy based on using multiple sources vs. average mapping accuracy using a single source, based on using 3 days of unlabeled target data. The mapping accuracies varies from space to space, depending on the consistency of activities in target space, as well as the similarity between the source and target spaces. It should be noted that some activities might not be present in all spaces, such as working or housekeeping. The same applies for lack of certain spaces in different apartments, such as laundry room or workspace. We noted that transfer between apartments that have a more similar layout and functional structure is more satisfactory.

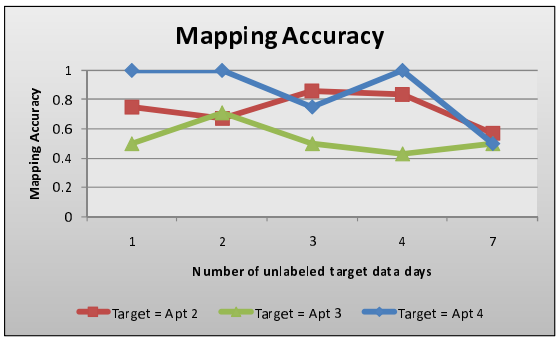


Figure 5: Mapping accuracy in several different settings.

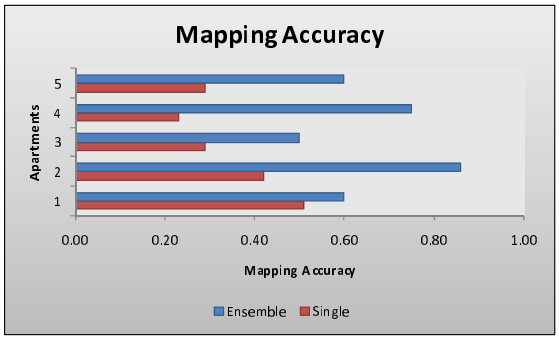


Figure 6: Mapping accuracy in several different settings.

We tested two of our own activity recognition algorithms on the transferred labeled data. The first algorithm is a nearest neighborhood (NN) algorithm based on the similarity measure in Equation 2. The second algorithm is a standard hidden Markov model (HMM). The models almost performed the same with the nearest neighborhood algorithm sometimes slightly outperforming HMM due to its use of temporal and spatial features. Using the embedded labels we define the recognition rate as the percentage of sensor events predicted with the correct label. Figure 7 shows NN's recognition rate based for apartment 1 as the target apartment using 0 and 1 day of labeled target data. Figure 8 shows both NN and HMM recognition rate for apartment 1 as the target apartment. Our results show that despite using little to no labeled target data, and having different layouts, schedules and different activities, both algorithms still perform recognition in a target space using data from a source space.

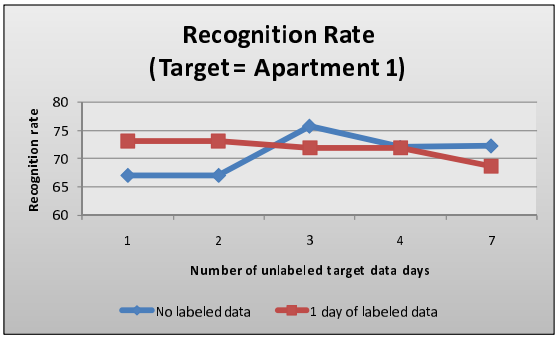


Figure 7: Nearest neighborhood's recognition rate based on mapping to apartment 1 using 0 and 1 day of labeled target data.

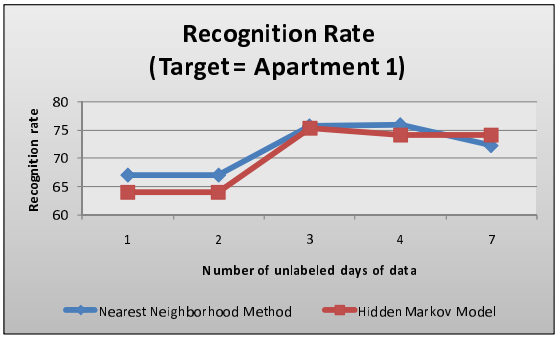


Figure 8: Recognition rate based on mapping to apartment 1 for nearest neighborhood and HMM.

4. CONCLUSIONS AND FUTURE WORK

This paper introduces a method of transferring learned activities from several different physical spaces to a target physical space. Transferring activities to a target space allows us to avoid the time consuming task of data annotation for each new physical space and to achieve a more accelerated deployment process. Using data from multiple source spaces allows us to be able to map from a more diverse set of activities to a target space by using an ensemble method. Our experiment results show that it is possible to recognize activities using no labeled data from the target spaces, and despite the fact that the apartment layouts and residents schedules are different. In the future, we intend to combine this method with adaptive and active learning methods in order to be able to enhance the results over time. We also want to develop algorithms that can map activities from environments with totally different functionalities, such as from a workplace to a residential space. We also intend to find methods for selecting the best subset of physical source

spaces among a large set of source spaces, in order to improve the system's efficiency.

5. ACKNOWLEDGEMENT

The authors would like to thank Brian Thomas for developing the visualizer software and making available the activity distribution diagrams.

6. REFERENCES

[1] G. Abowd and E. Mynatt. *Smart Environments: Technology, Protocols, and Applications*, chapter Designing for the human experience in smart environments., pages 153–174. Wiley, 2004.

[2] O. Brdiczka, J. Maisonnasse, and P. Reignier. Automatic detection of interaction groups. In *Proceedings of the 7th international conference on Multimodal interfaces*, pages 32–36, 2005.

[3] F. Can. Incremental clustering for dynamic information processing. *ACM Transactions on Information Systems*, 11(2):143–164, 1993.

[4] R. Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

[5] D. Cook and S. Das. *Smart Environments: Technology, Protocols and Applications*. Wiley Series on Parallel and Distributed Computing. Wiley-Interscience, 2004.

[6] D. Cook, M. Youngblood, I. Heierman, E.O., K. Gopalratnam, S. Rao, A. Litvin, and F. Khawaja. Mavhome: an agent-based smart home. In *Proceedings of the First IEEE International Conference on Pervasive Computing and Communications*, pages 521–524, March 2003.

[7] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *The Royal Statistical Society*, 39(1):1–38, 1977.

[8] T. G. Dietterich. Ensemble methods in machine learning. In *MCS '00: Proceedings of the First International Workshop on Multiple Classifier Systems*, pages 1–15, 2000.

[9] F. Doctor, H. Hagras, and V. Callaghan. A fuzzy embedded agent-based approach for realizing ambient intelligence in intelligent inhabited environments. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 35(1):55–65, Jan. 2005.

[10] T. Gu, Z. Wu, X. Tao, H. Pung, , and J. Lu. epsicar: An emerging patterns based approach to sequential, interleaved and concurrent activity recognition. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communication*, 2009.

[11] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Machine Learning Research*, 3:1157–1182, 2003.

[12] S. Helal, W. Mann, H. El-Zabadani, J. King, Y. Kaddoura, and E. Jansen. The gator tech smart house: A programmable pervasive space. *Computer*, 38(3):50–60, 2005.

[13] T. Inomata, F. Naya, N. Kuwahara, F. Hattori, and K. Kogure. Activity recognition from interactions with objects using dynamic bayesian network. In *Casemans '09: Proceedings of the 3rd ACM International Workshop on Context-Awareness for Self-Managing Systems*, pages 39–42, 2009.

[14] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991.

[15] L. Liao, D. Fox, and H. Kautz. Location-based activity recognition using relational markov networks. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 773–778, 2005.

[16] U. Maurer, A. Smailagic, D. P. Siewiorek, and M. Deisher. Activity recognition and monitoring using multiple sensors on different body positions. In *BSN '06: Proceedings of the International Workshop on Wearable and Implantable Body Sensor Networks*, pages 113–116, 2006.

[17] S. J. Pan and Q. Yang. A survey on transfer learning. Technical Report HKUST-CS08-08, Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong, China, November 2008.

[18] M. Philipose, K. Fishkin, M. Perkowitz, D. Patterson, D. Fox, H. Kautz, and D. Hahnel. Inferring activities from interactions with objects. *IEEE Pervasive Computing*, 3(4):50–57, Oct.-Dec. 2004.

[19] R. Raina, A. Y. Ng, and D. Koller. Constructing informative priors using transfer learning. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 713–720, 2006.

[20] P. Rashidi and D. J. Cook. An adaptive sensor mining framework for pervasive computing applications. In *International Workshop on Knowledge Discovery from Sensor Data(Sensor-KDD 2008)*, pages 41–49, 2008.

[21] P. Rashidi and D. J. Cook. the resident in the loop: Adapting the smart home to the user. *IEEE Transactions on Systems, Man, and Cybernetics journal, Part A*, 39(5):949–959, September 2009.

[22] P. Rashidi and D. J. Cook. Transferring learned activities in smart environments. In *5th International Conference on Intelligent Environments*, volume 2 of *Ambient Intelligence and Smart Environments*, pages 185–192, 2009.

[23] V. Rialle, C. Ollivet, C. Guigui, and C. Hervé. What do family caregivers of alzheimer's disease patients desire in smart home technologies? contrasted results of a wide survey. *Methods of Information in Medicine*, 47:63–9, 2008.

[24] T. van Kasteren, G. Englebienne, and B. Krose. Recognizing activities in multiple contexts using transfer learning. In *AAAI AI in Eldercare Symposium*, 2008.

[25] C. Wren and E. Munguia-Tapia. Toward scalable activity recognition for sensor networks. In *Proceedings of the Workshop on Location and Context-Awareness*, pages 218–235, 2006.

[26] V. W. Zheng, D. H. Hu, and Q. Yang. Cross-domain activity recognition. In *Ubicomp '09: Proceedings of the 11th international conference on Ubiquitous computing*, pages 61–70, 2009.